UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P O Box 1450
Alexandria, Virginia 22313-1450
www uspto gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

27927 7590 09/03/2010

RICHARD AUCHTERLONIE
NOVAK DRUCE & QUIGG, LLP
1000 LOUISIANA
53RD FLOOR
HOUSTON, TX 77002

EXAMINER	
BIBBEE, JARED M	
ART UNIT	PAPER NUMBER
2161	

DATE MAILED: 09/03/2010

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/668,467	09/23/2003	Sachin Mullick	10830.0100.NPUS00	2942

TITLE OF INVENTION: MULTI-THREADED WRITE INTERFACE AND METHODS FOR INCREASING THE SINGLE FILE READ AND WRITE THROUGHPUT OF A FILE SERVER

APPLN. TYPE	SMALL ENTITY	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	NO	$1510	$300	$0	$1810	12/03/2010

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE DOES NOT REFLECT A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE IN THIS APPLICATION. IF AN ISSUE FEE HAS PREVIOUSLY BEEN PAID IN THIS APPLICATION (AS SHOWN ABOVE), THE RETURN OF PART B OF THIS FORM WILL BE CONSIDERED A REQUEST TO REAPPLY THE PREVIOUSLY PAID ISSUE FEE TOWARD THE ISSUE FEE NOW DUE.

HOW TO REPLY TO THIS NOTICE:

I. Review the SMALL ENTITY status shown above.

If the SMALL ENTITY is shown as YES, verify your current SMALL ENTITY status:

A. If the status is the same, pay the TOTAL FEE(S) DUE shown above.

B. If the status above is to be removed, check box 5b on Part B - Fee(s) Transmittal and pay the PUBLICATION FEE (if required) and twice the amount of the ISSUE FEE shown above, or

If the SMALL ENTITY is shown as NO:

A. Pay TOTAL FEE(S) DUE shown above, or

B. If applicant claimed SMALL ENTITY status before, or is now claiming SMALL ENTITY status, check box 5a on Part B - Fee(s) Transmittal and pay the PUBLICATION FEE (if required) and 1/2 the ISSUE FEE shown above.

II. PART B - FEE(S) TRANSMITTAL, or its equivalent, must be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted. If an equivalent of Part B is filed, a request to reapply a previously paid issue fee must be clearly made, and delays in processing may occur due to the difficulty in recognizing the paper as an equivalent of Part B.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Mail Stop ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Utility patents issuing on applications filed on or after Dec. 12, 1980 may require payment of maintenance fees. It is patentee's responsibility to ensure timely payment of maintenance fees when due.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), to: Mail **Mail Stop ISSUE FEE**
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450
or Fax (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

27927 7590 09/03/2010

RICHARD AUCHTERLONIE
NOVAK DRUCE & QUIGG, LLP
1000 LOUISIANA
53RD FLOOR
HOUSTON, TX 77002

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission
I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being facsimile transmitted to the USPTO (571) 273-2885, on the date indicated below.

(Depositor's name)
(Signature)
(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/668,467	09/23/2003	Sachin Mullick	I0830.0100.NPUS00	2942

TITLE OF INVENTION: MULTI-THREADED WRITE INTERFACE AND METHODS FOR INCREASING THE SINGLE FILE READ AND WRITE THROUGHPUT OF A FILE SERVER

APPLN. TYPE	SMALL ENTITY	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	NO	$1510	$300	$0	$1810	12/03/2010

EXAMINER	ART UNIT	CLASS-SUBCLASS
BIBBEE, JARED M	2161	707-704000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).
☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.
☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-02 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) the names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.
1 ____
2 ____
3 ____

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document has been filed for recordation as set forth in 37 CFR 3.11. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE (B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ☐ Individual ☐ Corporation or other private group entity ☐ Government

4a. The following fee(s) are submitted:
☐ Issue Fee
☐ Publication Fee (No small entity discount permitted)
☐ Advance Order - # of Copies ____

4b. Payment of Fee(s): **(Please first reapply any previously paid issue fee shown above)**
☐ A check is enclosed.
☐ Payment by credit card. Form PTO-2038 is attached.
☐ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment, to Deposit Account Number ____ (enclose an extra copy of this form).

5. **Change in Entity Status** (from status indicated above)
☐ a. Applicant claims SMALL ENTITY status. See 37 CFR 1.27.
☐ b. Applicant is no longer claiming SMALL ENTITY status. See 37 CFR 1.27(g)(2).

NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.

Authorized Signature ____ Date ____

Typed or printed name ____ Registration No. ____

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P O Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/668,467	09/23/2003	Sachin Mullick	10830.0100.NPUS00	2942

27927 7590 09/03/2010

RICHARD AUCHTERLONIE
NOVAK DRUCE & QUIGG, LLP
1000 LOUISIANA
53RD FLOOR
HOUSTON, TX 77002

EXAMINER	
BIBBEE, JARED M	
ART UNIT	PAPER NUMBER
2161	

DATE MAILED: 09/03/2010

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)
(application filed on or after May 29, 2000)

The Patent Term Adjustment to date is 904 day(s). If the issue fee is paid on the date that is three months after the mailing date of this notice and the patent issues on the Tuesday before the date that is 28 weeks (six and a half months) after the mailing date of this notice, the Patent Term Adjustment will be 904 day(s).

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Adjustment is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at 1-(888)-786-0101 or (571)-272-4200.

Notice of Allowability	**Application No.** 10/668,467	**Applicant(s)** MULLICK ET AL.
	Examiner JARED M. BIBBEE	**Art Unit** 2161

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☒ This communication is responsive to *Appeal Brief filed 6/13/2010*.
2. ☒ The allowed claim(s) is/are *1-12, 17-28, 32-46, 51-54, 58-62, 65, 67-68, 71 and 73 (re-numbered 1-53)*.
3. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
   a) ☐ All b) ☐ Some* c) ☐ None of the:
   1. ☐ Certified copies of the priority documents have been received.
   2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
   3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____.

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

4. ☐ A SUBSTITUTE OATH OR DECLARATION must be submitted. Note the attached EXAMINER'S AMENDMENT or NOTICE OF INFORMAL PATENT APPLICATION (PTO-152) which gives reason(s) why the oath or declaration is deficient.
5. ☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
   (a) ☐ including changes required by the Notice of Draftsperson's Patent Drawing Review (PTO-948) attached
   1) ☐ hereto or 2) ☐ to Paper No./Mail Date _____.
   (b) ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____.

   Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).
6. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. ☐ Notice of References Cited (PTO-892)
2. ☐ Notice of Draftperson's Patent Drawing Review (PTO-948)
3. ☐ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date _____
4. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material
5. ☐ Notice of Informal Patent Application
6. ☐ Interview Summary (PTO-413), Paper No./Mail Date _____ .
7. ☒ Examiner's Amendment/Comment
8. ☒ Examiner's Statement of Reasons for Allowance
9. ☐ Other _____.

/Jared M Bibbee/ Examiner, Art Unit 2161	/Apu M Mofiz/ Supervisory Patent Examiner, Art Unit 2161

EXAMINER'S AMENDMENT

Examiner was given permission to amend the application based on the proposed amendment submitted by Applicant's representative on 8/26/2010 and also via telephone on 8/25/2010.

The application has been amended as follows:

1. (Previously presented) A method of operating a network file server computer for providing clients with concurrent write access to a file in data storage, the method comprising the network file server computer responding to a concurrent write request from a client by:

(a) obtaining a lock for the file; and then

(b) preallocating a metadata block for the file; and then

(c) releasing the lock for the file; and then

(d) asynchronously writing to the file; and then

(e) obtaining the lock for the file; and then

(f) committing the metadata block to the file in the data storage; and then

(g) releasing the lock for the file.

2. (Previously presented) The method as claimed in claim 1, wherein the file further includes a hierarchy of blocks including an inode block of metadata, data blocks of file data, and indirect blocks of metadata, and wherein the metadata block for the file is an indirect block of metadata.

3. (Previously presented) The method as claimed in claim 2, which further includes copying data from an original indirect block of the file to the metadata block for the file, the original indirect block of the file having been shared between the file and a read-only version of the file.

4. (Previously presented) The method as claimed in claim 1, which further includes concurrent writing for more than one client to the metadata block for the file.

5. (Previously presented) The method as claimed in claim 1, wherein the asynchronous writing to the file includes a partial write to a new block that has been copied at least in part from an original block of the file, and wherein the method further includes checking a partial block conflict queue for a conflict with a concurrent write to the new block, and upon failing to find an indication of a conflict with a concurrent write to the new block, preallocating the new block, copying at least a portion of the original block of the file to the new block, and performing the partial write to the new block.

6. (Previously presented) The method as claimed in claim 1, wherein the asynchronous writing to the file includes a partial write to a new block that has been copied at least in part from an original block of the file, and wherein the method further includes checking a partial block conflict queue for a conflict with a concurrent write to the new block, and upon finding an indication of a conflict with a concurrent write to the new block, waiting until resolution of the

conflict with the concurrent write to the new block, and then performing the partial write to the new block.

7. (Previously presented) The method as claimed in claim 6, which further includes placing a request for the partial write in a partial write wait queue upon finding an indication of a conflict with a concurrent write to the new block, and performing the partial write upon servicing the partial write wait queue.

8. (Previously presented) The method as claimed in claim 1, which further includes checking an input-output list for a conflicting prior concurrent access to the file, and upon finding a conflicting prior concurrent access to the file, suspending the asynchronous writing to the file until the conflicting prior concurrent access to the file is no longer conflicting.

9. (Previously presented) The method as claimed in claim 8, which further includes providing a sector-level granularity of byte range locking for concurrent write access to the file by the suspending of the asynchronous writing to the file until the conflicting prior concurrent access is no longer conflicting.

10. (Previously presented) The method as claimed in claim 1, which further includes writing the metadata block to a log in storage of the network file server computer for committing the metadata block for the file.

11. (Previously presented) The method as claimed in claim 1, which further includes gathering together preallocated metadata blocks for a plurality of client write requests to the file, and committing together the preallocated metadata blocks for the plurality of client write requests to the file by obtaining the lock for the file, committing the gathered preallocated metadata blocks for the plurality of client write requests to the file, and then releasing the lock for the file.

12. (Previously presented) The method as claimed in claim 1, which further includes checking whether a previous commit is in progress after asynchronously writing to the file and before obtaining the lock for the file for committing the metadata block to the file, and upon finding that a previous commit is in progress, placing a request for committing the metadata block to the file on a staging queue for the file.

Claims 13-16 (Canceled).

17. (Currently amended) The method as claimed in claim [[15]] 11, wherein the network file server computer includes disk storage containing a file system, and a file system cache storing data of blocks of the file, and the method further includes the network file server computer responding to concurrent write requests by writing new data for specified blocks of the file to the disk storage without writing the new data for the specified blocks of the file to the file system cache, and invalidating the specified blocks of the file in the file system cache.

18. (Previously presented) The method as claimed in claim 17, which further includes the network file server computer responding to read requests for file blocks not found in the file system cache by reading the file blocks from the file system in disk storage and then checking whether the file blocks have become stale due to concurrent writes to the file blocks, and writing to the file system cache a file block that has not become stale, and not writing to the file system cache a file block that has become stale.

19. (Previously presented) The method as claimed in claim 18, which further includes the network file server computer checking a read-in-progress flag for a file block upon finding that the file block is not in the file system cache, and upon finding that the read-in-progress flag indicates that a prior read of the file block is in progress from the file system in the disk storage, waiting for completion of the prior read of the file block from the file system in the disk storage, and then again checking whether the file block is in the file system cache.

20. (Previously presented) The method as claimed in claim 18, which further includes the network file server computer setting a read-in-progress flag for a file block upon finding that the file block is not in the file system cache and then beginning to read the file block from the file system in disk storage, clearing the read-in-progress flag upon writing to the file block on disk, and inspecting the read-in-progress flag to determine whether the file block has become stale due to a concurrent write to the file block.

21. (Previously presented) The method as claimed in claim 18, which further includes the network file server computer maintaining a generation count for each read of a file block from the file system in the disk storage in response to a read request for a file block that is not in the file system cache, and checking whether a file block having been read from the file system in the disk storage has become stale by checking whether the generation count for the file block having been read from the file system is the same as the generation count for the last read request for the same file block.

22. (Currently amended) The method as claimed in claim [[15]] 11, which further includes processing multiple concurrent read and write requests by pipelining the requests through a first processor and a second processor, the first processor performing metadata management for the multiple concurrent read and write requests, and the second processor performing asynchronous reads and writes for the multiple concurrent read and write requests.

23. (Currently amended) The method as claimed in claim [[15]] 11, which further includes serializing the reads by delaying access for each read to a block that is being written to by a prior, in-progress write until completion of the write to the block that is being written to by the prior, in-progress write.

24. (Currently amended) The method as claimed in claim [[15]] 11, which further includes serializing the writes by delaying access for each write to a block that is being accessed by a

prior, in-progress read or write until completion of the read or write to the block that is being accessed by the prior, in-progress read or write.

25. (Currently amended) ~~A method of operating a network file server computer for providing clients with concurrent read and write access to a file in data storage, the method comprising the network file server computer responding to a concurrent write request from a client by:~~

~~(a) preallocating a metadata block for the file; and then~~

~~(b) asynchronously writing to the file; and then~~

~~(c) committing the metadata block to the file in the data storage~~;

The method as claimed in claim 1, wherein the network file server computer includes disk storage containing a file system, and a file system cache storing data of blocks of the file, and the method includes the network file server computer responding to concurrent write requests by writing new data for specified blocks of the file to the disk storage without writing the new data for the specified blocks of the file to the file system cache, and invalidating the specified blocks of the file in the file system cache, and

which includes the network file server computer responding to read requests for file blocks not found in the file system cache by reading the file blocks from the file system in disk storage and then checking whether the file blocks have become stale due to concurrent writes to the file blocks, and writing to the file system cache a file block that has not become stale, and not writing to the file system cache a file block that has become stale.

26. (Previously presented) The method as claimed in claim 25, which further includes the network file server computer checking a read-in-progress flag for a file block upon finding that the file block is not in the file system cache, and upon finding that the read-in-progress flag indicates that a prior read of the file block is in progress from the file system in the disk storage, waiting for completion of the prior read of the file block, and then again checking whether the file block is in the file system cache.

27. (Previously presented) The method as claimed in claim 25, which further includes the network file server computer setting a read-in-progress flag for a file block upon finding that the file block is not in the file system cache and then beginning to read the file block from the file system in disk storage, clearing the read-in-progress flag upon writing to the file block on disk, and inspecting the read-in-progress flag to determine whether the file block has become stale due to a concurrent write to the file block.

28. (Previously presented) The method as claimed in claim 25, which further includes the network file server computer maintaining a generation count for each read of a file block from the file system in the disk storage in response to a read request for a file block that is not in the file system cache, and checking whether a file block having been read from the file system in the disk storage has become stale by checking whether the generation count for the file block having been read from the file system is the same as the generation count for the last read request for the same file block.

Claims 29-31 (Canceled).

32. (Previously presented) A method of operating a network file server computer for providing clients with concurrent write access to a file in data storage, the method comprising the network file server computer responding to a concurrent write request from a client by executing a write thread, execution of the write thread including:

(a) obtaining an allocation mutex for the file; and then

(b) preallocating new metadata blocks that need to be allocated for writing to the file; and then

(c) releasing the allocation mutex for the file; and then

(d) issuing asynchronous write requests for writing to the file;

(e) waiting for callbacks indicating completion of the asynchronous write requests; and then

(f) obtaining the allocation mutex for the file; and then

(g) committing the preallocated metadata blocks to the file in the data storage; and then

(h) releasing the allocation mutex for the file.

33. (Original) A network file server comprising storage for storing a file, and at least one processor coupled to the storage for providing clients with concurrent write access to the file, wherein the network file server is programmed for responding to a concurrent write request from a client by:

(a) obtaining a lock for the file; and then

(b) preallocating a metadata block for the file; and then

(c) releasing the lock for the file; and then

(d) asynchronously writing to the file; and then

(e) obtaining the lock for the file; and then

(f) committing the metadata block to the file; and then

(g) releasing the lock for the file.

34. (Previously presented) The network file server as claimed in claim 33, wherein the file further includes a hierarchy of blocks including an inode block of metadata, data blocks of file data, and indirect blocks of metadata, and wherein the metadata block for the file is an indirect block of metadata.

35. (Previously presented) The network file server as claimed in claim 34, which is further programmed for copying data from an original indirect block of the file to the metadata block for the file, the original indirect block of the file having been shared between the file and a read-only version of the file.

36. (Previously presented) The network file server as claimed in claim 33, which is further programmed for concurrent writing for more than one client to the metadata block for the file.

37. (Previously presented) The network file server as claimed in claim 33, which further includes a partial block conflict queue for indicating a concurrent write to a new block that is

being copied at least in part from an original block of the file, and wherein the network file server is further programmed to respond to a client request for a partial write to the new block by checking the partial block conflict queue for a conflict, and upon failing to find an indication of a conflict, preallocating the new block, copying at least a portion of the original block of the file to the new block, and performing a partial write to the new block.

38. (Previously presented) The network file server as claimed in claim 33, which further includes a partial block conflict queue for indicating a concurrent write to a new block that is being copied at least in part from an original block of the file, and wherein the network file server is further programmed to respond to a client request for a partial write to the new block by checking the partial block conflict queue for a conflict, and upon finding an indication of a conflict, waiting until resolution of the conflict with the concurrent write to the new block, and then performing the partial write to the new block.

39. (Previously presented) The network file server as claimed in claim 38, which further includes a partial write wait queue, and wherein the network file server is further programmed for placing a request for the partial write in the partial write wait queue upon finding an indication of a conflict, and performing the partial write upon servicing the partial write wait queue.

40. (Previously presented) The network file server as claimed in claim 33, which is further programmed for maintaining an input-output list of concurrent reads and writes to the file, and

when writing to the file, for checking the input-output list for a conflicting prior concurrent read or write access to the file, and upon finding a conflicting prior concurrent read or write access to the file, suspending the asynchronous writing to the file until the conflicting prior concurrent read or write access to the file is no longer conflicting.

41. (Previously presented) The network file server as claimed in claim 40, which is further programmed so that the suspending of the asynchronous writing to the file until the conflicting prior concurrent read or write access to the file is no longer conflicting provides a sector-level granularity of byte range locking for concurrent write access to the file.

42. (Previously presented) The network file server as claimed in claim 33, which is further programmed for maintaining an input-output list of concurrent reads and writes to the file, and when reading from the file, for checking the input-output list for a conflicting prior concurrent write access to the file, and upon finding a conflicting prior concurrent write access to the file, suspending the reading to the file until the conflicting prior concurrent write access to the file is no longer conflicting.

43. (Previously presented) The network file server as claimed in claim 42, which is further programmed so that the suspending of the reading to the file until the conflicting prior concurrent write access to the file is no longer conflicting provides a sector-level granularity of byte range locking for concurrent read access to the file.

44. (Previously presented) The network file server as claimed in claim 33, which is further programmed for committing the metadata block for the file by writing the metadata block to a log in the storage.

45. (Previously presented) The network file server as claimed in claim 33, which is further programmed for gathering together preallocated metadata blocks for a plurality of client requests for write access to the file, and committing together the preallocated metadata blocks for the plurality of client requests for access to the file by obtaining the lock for the file, committing the gathered preallocated metadata blocks for the plurality of client requests for write access to the file, and then releasing the lock for the file.

46. (Previously presented) The network file server as claimed in claim 33, which further includes a staging queue for the file, and which is further programmed for checking whether a previous commit is in progress after asynchronously writing to the file and before obtaining the lock for the file for committing the metadata block to the file, and upon finding that a previous commit is in progress, placing a request for committing the metadata block to the file on the staging queue for the file.

Claims 47-50 (Canceled).

51. (Currently amended) ~~A network file server comprising disk storage containing~~

The network file server as claimed in claim 33, further comprising a file system, and a file system cache storing data of blocks of a file in the file system, ~~wherein the network file server is programmed for responding to a concurrent write request from a client by:~~

~~(a) preallocating a metadata block for the file; and then~~

~~(b) asynchronously writing to the file; and then~~

~~(c) committing the metadata block to the file;~~

wherein the network file server is further programmed for responding to concurrent write requests by writing new data for specified blocks of the file to the disk storage without writing the new data for the specified blocks of the file to the file system cache, and invalidating the specified blocks of the file in the file system cache, and

wherein the network file server is programmed for responding to concurrent read requests for file blocks not found in the file system cache by reading the file blocks from the file system in disk storage and then checking whether the file blocks have become stale due to concurrent writes to the file blocks, and writing to the file system cache a file block that has not become stale, and not writing to the file system cache a file block that has become stale.

52. (Previously presented) The network file server as claimed in claim 51, which is further programmed for checking a read-in-progress flag for a file block upon finding that the file block is not in the file system cache, and upon finding that the read-in-progress flag indicates that a prior read of the file block is in progress from the file system in the disk storage, waiting for completion of the prior read of the file block, and then again checking whether the file block is in the file system cache.

53. (Previously presented) The network file server as claimed in claim 51, which is further programmed for setting a read-in-progress flag for a file block upon finding that the file block is not in the file system cache and then beginning to read the file block from the file system in disk storage, clearing the read-in-progress flag upon writing to the file block on disk, and inspecting the read-in-progress flag to determine whether the file block has become stale due to a concurrent write to the file block.

54. (Previously presented) The network file server as claimed in claim 51, which is further programmed for maintaining a generation count for each read of a file block from the file system in the disk storage in response to a read request for a file block that is not in the file system cache, and checking whether a file block having been read from the file system in the disk storage has become stale by checking whether the generation count for the file block having been read from the file system is the same as the generation count for the last read request for the same file block.

Claims 55-57 (Cancelled).

58. (Original) A network file server comprising storage for storing a file, and at least one processor coupled to the storage for providing clients with concurrent write access to the file, wherein the network file server is programmed with a write thread for responding to a concurrent write request from a client by:

(a) obtaining an allocation mutex for the file; and then

(b) preallocating new metadata blocks that need to be allocated for writing to the file; and then

(c) releasing the allocation mutex for the file; and then

(d) issuing asynchronous write requests for writing to the file;

(e) waiting for callbacks indicating completion of the asynchronous write requests; and then

(f) obtaining the allocation mutex for the file; and then

(g) committing the preallocated metadata blocks; and then

(h) releasing the allocation mutex for the file.

59. (Previously presented) The network file server as claimed in claim 58, which further includes an uncached write interface, a file system cache and a cached read-write interface, and wherein the uncached write interface bypasses the file system cache for sector-aligned write operations.

60. (Previously presented) The network file server as claimed in claim 59, wherein the network file server is further programmed to invalidate cache blocks in the file system cache including sectors being written to by the uncached write interface.

61. (Currently amended) ~~A network file server comprising storage for storing a file, and at least one processor coupled to the storage for providing clients with concurrent write access to~~

~~the file, wherein the network file server is programmed for responding to a concurrent write request from a client by:~~

~~(a) preallocating a block for writing to the file;~~

~~(b) asynchronously writing to the file; and then~~

~~(c) committing the preallocated block;~~

The network file server as claimed in claim 33, wherein the network file server also includes an uncached write interface, a file system cache, and a cached read-write interface, wherein the uncached write interface bypasses the file system cache for sector-aligned write operations, and the network file server is programmed to invalidate cache blocks in the file system cache including sectors being written to by the uncached write interface.

62. (Previously presented) The method as claimed in claim 1, which further includes a final step of returning to said client an acknowledgement of the writing to the file.

Claims 63-64 (Canceled).

65. (Previously presented) The method as claimed in claim 25, which further includes a final step of returning to said client an acknowledgement of the writing to the file.

Claim 66 (Canceled).

67. (Previously presented) The method as claimed in claim 32, which further includes a final step of returning to said client an acknowledgement of the writing to the file.

68. (Previously presented) The method as claimed in claim 1, which further includes a final step of saving the file in disk storage of the network file server.

Claims 69-70 (Canceled).

71. (Previously presented) The method as claimed in claim 25, which further includes a final step of saving the file in the disk storage.

Claim 72 (Canceled).

73. (Previously presented) The method as claimed in claim 32, which further includes a final step of saving the file in disk storage of the network file server.

Reasons for Allowance

The following is an examiner's statement of reasons for allowance:

With regards to independent claims 1, 32, 33, and 58, these claims contain limitations that overcome the best possible prior art. The best prior art in this case is Burns et al (US 6,925,515 B2), herein after "Burns", Marcotte (US 6,449, 614 B1), herein after “Marcotte”, and Xu et al (US 6,324,581 B1), herein after “Xu”.

Burns, Marcotte, and Xu fail to disclose/teach the limitations of:

- obtaining a lock for the file; and then
- preallocating a metadata block for the file; and then
- releasing the lock for the file; and then
- asynchronously writing to the file; and then
- obtaining the lock for the file; and then
- committing the metadata block to the file in the data storage; and then
- releasing the lock for the file.

Claims 2-12, 17-28, 34-46, 51-54, 59-62, 65, 67-68, 71 and 73 depend from claims 1, 32, 33, and 58 and are allowable for at least the same reasons as set forth above.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to JARED M. BIBBEE whose telephone number is (571)270-1054. The examiner can normally be reached on IFP.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Apu Mofiz can be reached on 571-272-4080. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Jared M Bibbee/
Examiner, Art Unit 2161

/Apu M Mofiz/

Supervisory Patent Examiner, Art Unit 2161